A NOVEL LANE FEATURE EXTRACTION ALGORITHM IMPLEMENTED ON THE TMS320DM6437 DSP PLATFORM

Yifei Wang, Naim Dahnoun and Alin Achim

Department of Electronic and Electrical Engineering
University of Bristol
Bristol BS8 1UB

ABSTRACT

This paper presents an implementation of a novel lane edge feature extraction algorithm based on digital interpolation. The method is based on the observation that by zooming towards the vanishing point, the positions and the sizes of the lanes will not change significantly while other objects will. Comparing the zoomed image with the original image allows us to remove most of the unwanted features from the lane feature map. The proposed algorithm shows outstanding performance on extracting features belonging solely to the lanes embedded in severe noise environment. The algorithm runs in real-time on the TMS320DM6437 DSP platform from Texas Instruments. The system implemented can achieve at least 23 fps, without performing any code optimisation but only considering memory management.

Index Terms— Lane detection, feature extraction, digital interpolation, DSP implementation

1. INTRODUCTION

Throughout the last two decades, a significant amount of research has been carried out in the area of lane detection. The input for most lane detection systems is the video captured by a forward pointing CCD camera. After the lane boundaries in the video are extracted, the lateral offset of the vehicle between the lanes can be calculated. Furthermore, the lane positions and shapes are useful for path planning.

A large number of existing lane detection systems have been proposed. Most of these systems focused on the detection of the lanes instead of the lane feature extraction. Some of the systems based on various lanes features are reviewed here: [1] introduced the Three-Feature Based Automatic Lane Detection Algorithm (TFALDA). This algorithm searches for lines in predefined search regions. After that, the line intensity, position and the gradient direction are used for the detection stage. Although this system uses the global information of the lanes, the algorithm tries to find all the straight lines. The lines corresponding to the lanes are also chosen at the detection stage. [2] presented the Likelihood Of Image Shape (LOIS) lane detection system. This system uses the deformable template approach based on the image gradient to allocate the lane boundaries. In this case, the detection stage is time consuming because all the pixels are taken into consideration. LANA system [3] is similar to LOIS system at the detection stage but uses frequency features of the lanes instead of edges. The system uses additional hardware to perform DCT operations and the detection time is reduced by involving only diagonally orientated features. [4] uses a filter to extract dark-bright-dark patterns and emphasis the lane features. The algorithm also uses frequency analysis method to classify continuous, segmented or merged lane markings. [5] introduced a system that uses the B-spline lane model, as well as the Canny/Hough Estimation of Vanishing Points (CHEVP) of the horizontally segmented lanes. The control points are positioned by the snake algorithm based on the gradient vector field (GVF) [6]. Although the system is robust, the whole algorithm takes a Pentium 3 system more than 3s to process a 240×256 pixels image. [7] uses texture anisotropy fields as features to segment the lane from the background. For most of the existing systems, the global shape information is only included in the detection stage but not in the feature extraction.

A model-based lane detection system normally consists of four parts: lane modelling (straight lines, parabolas etc.), feature extraction (edge, texture, frequency etc.), detection (Hough transform, deformable template matching etc.) and tracking (Kalman filtering, particle filtering etc.). In this paper, we propose an algorithm that focuses on the feature extraction stage. By using the characteristics of the lanes and their global shape information, the proposed feature extraction algorithm is able to verify whether the extracted features belong to the lanes. First, an appropriate area around the vanishing point is selected and interpolated back to the original image size. While superimposing the interpolated image with the original image, the lanes on both images will overlap but not the other objects. Based on this observation, the edges of the interpolated images are compared with the original image

A. Achim was supported in part by the European Science Foundation through the COST Action TU0702

978-1-4244-3298-1/09/$25.00 ©2009 IEEE

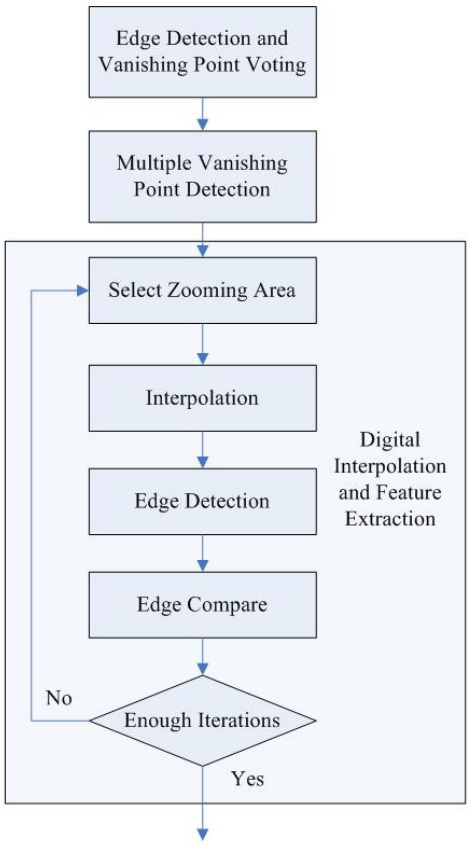


Fig. 1: Overall algorithm block diagram.

edge map, and most of the irrelevant edges can be removed. The system block diagram is shown in Fig. 1.

This paper describes the implementation of the proposed lane feature extraction algorithm in details. The algorithm is implemented on the TMS320DM6437 DSP platform from Texas Instrument. This high-performance fixed-point Digital Media Processor (DM6437) clocked at 600 MHz is specially suited for this type of applications.

Section 2 describes the theory behind the algorithm and concentrates on the effect of digital interpolating a lane image. Section 3 and 4 describe in details the vanishing point detection and feature extraction steps respectively. Section 5 introduces the TMS320DM6437 DSP platform. Section 6 describes the DSP implementation of the proposed algorithm. Section 7 illustrates some experimental results. Section 8 concludes the paper.

2. DIGITAL INTERPOLATION ON LANE IMAGES

The purpose of the algorithm is to find the features belonging solely to the lanes. While driving on a straight road with continuous lane marking, the positions of the lanes are not changing significantly over time from the driver's point of view.

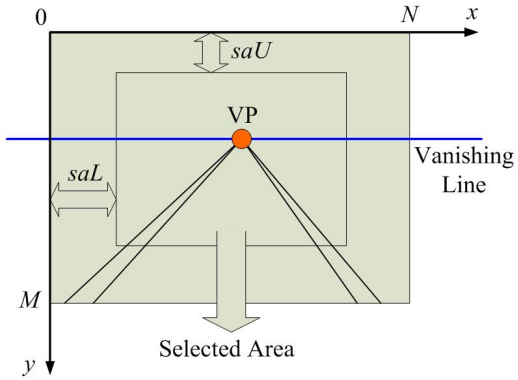


Fig. 2: Selected zooming area of an image.

This algorithm takes advantage of the above phenomenon and tries to find the non-moving features within the scene.

In order to do this, digital interpolation is applied. By carefully selecting a region of the image and interpolating this region back to the original image size, the driver's view can be simulated and the lanes on the interpolated and original images should overlap.

The first task is to select the appropriate area (SA) for interpolation. The vanishing point of the left and right lanes is where the vehicle is heading towards. After interpolation, the vanishing point should stay at the same position. As illustrated in Fig. 2, defining the position of vanishing point VP as (vpx, vpy), the total number of rows as M, and the number of columns as N, the width and height of the selected area can be calculated as:

$$\begin{cases} saN = zr \times N \\ saM = zr \times M \end{cases} \tag{1}$$

where zr is the zooming ratio and zr^2 is the ratio between the selected area and the original image area. The selection of the zooming area must follow the rules that:

$$\frac{vpx}{N - vpx} = \frac{vpx - saL}{saL + saN - vpx} \tag{2}$$

$$\frac{vpy}{M - vpy} = \frac{vpy - saU}{saU + saM - vpy} \tag{3}$$

where saL and saU are the position of the left and upper border of the selected area.

Subsequently, the selected area is interpolated back to the original size. This operation moves all points except the vanishing point to new positions, which are calculated as:

$$x_I(t+1) = vpx + (x_I(t) - vpx) \times \frac{1}{1 - zr} \tag{4}$$

$$y_I(t+1) = vpy + (y_I(t) - vpy) \times \frac{1}{1 - zr} \tag{5}$$

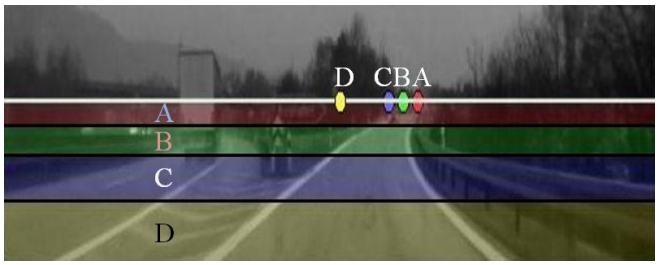


Fig. 3: Vanishing point detection result. The image is horizontally partitioned and each partition is labelled. The white horizontal line is the horizon or the vanishing line. The vanishing point of each region is labelled respectively.

where $x_I(t)$, $y_I(t)$ and $x_I(t+1)$, $y_I(t+1)$ represents the x and y coordinate of point I before and after the interpolation respectively. Assume a straight line:

$$y_I = ax_I + b \tag{6}$$

which passes through the vanishing point and I is a point on the line, substitute 6 into Equation 5 and rearrange, we get:

$$y_I(t+1) = ax_I(t+1) + b \tag{7}$$

Equation 7 indicates that the points on the lane boundaries will stay on the straight lines after interpolation.

3. VANISHING POINT DETECTION

A voting process is introduced to determine the vanishing point position. In this paper, we assume the road is flat. The vanishing line or the horizon can be calculated using the camera parameters. Since the vanishing point is on the vanishing line, a one dimensional accumulator is created. The size of the accumulator is $2N$ and it is able to represent possible positions from $-0.5N$ to $1.5N$ on the vanishing line according to Fig. 2.

First, Sobel edge detection is performed with a very small threshold. In our case, the threshold is between 10 and 40 for non-normalised gradient. For each of the detected edge, a line perpendicular to the edge orientation and passing through the edge is generated. The position of the intersection between this line and the vanishing line is an estimated vanishing point position. The corresponding element of the accumulator increments by $gm + 1$ where gm is the normalised gradient magnitude.

After the accumulation is finished, the data accumulated is smoothed by a low-pass filter. The element holding the maximum number of votes represents the corresponding vanishing point.

Up to this point, straight lanes have been assumed. In order to extent this to curved lanes, the image is partitioned into a number of horizontal bands and each of the image band holds an independent accumulator. Since the number of lane edges decreases with distance, the vanishing point of the lowest image band (nearest field) is detected first. The position of the vanishing point on the upper bands are detected based on the vanishing point position of the lower bands to reduce the detection error and the processing power. An example of multiple vanishing points detection is illustrated in Fig. 3. Finally the positions of the vanishing points can be tracked frame by frame to increase the detection accuracy.

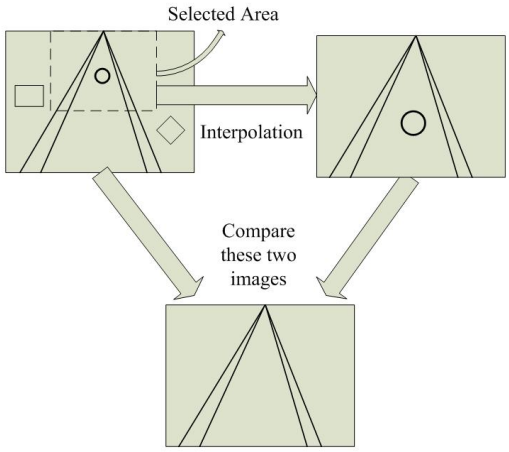


Fig. 4: Digital zooming and edge comparison.

4. LANE FEATURE EXTRACTION

In order to extract the lane features, the '*logical and*' operator is applied to the original image edge map and to the interpolated image edge map pixel by pixel. This means that if the interpolated edges overlap with the original image edges, these edges are preserved. Furthermore, the noise removal effect can be improved to constrain the orientation difference of the overlaid edges being within the range 0 to $\pi/2$ rads. Fig. 4 illustrates the digital interpolation along with the edge comparison process. Note that after interpolation, the position and size of the circle is changed but not the straight lines.

The zooming and edge comparison process with different zooming ratio is repeated until most of the unwanted features are removed. Based on experiments, 10 iterations are normally sufficient even under very severe conditions. During the edge comparison process, segmented lane markings will be shortened to zr times their original length. Although they will stay on the same line after interpolation, the upper part of the line will be erased. In this case, zr needs to be a large value. Experiments have showed that a value greater than 0.85 would be acceptable in most cases.

Since multiple vanishing points exist, each of the image bands zooms into the corresponding vanishing point. Also, the results of edge detection on each section are compared separately. Example feature extraction results in comparison

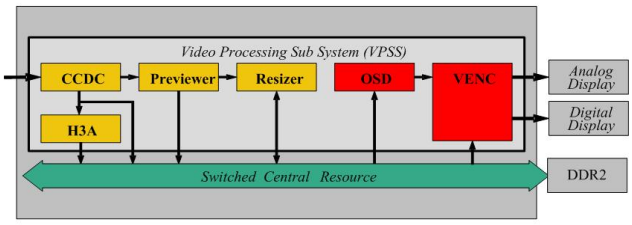


Fig. 5: Video Processing Sub System (VPSS).

with traditional gradient maps are shown in Fig. 8.

5. TMS320DM6437 DSP PLATFORM

Automobiles are now using a large number of DSP processors for applications ranging from automatic navigation, driver information to auto entertainment and therefore the processing power, data format, power consumption, cost and size are becoming important factors for selecting the right DSP.

In this application, the fixed-point TMSDM6437 Digital Media Processor has been selected. The processor can be clocked at a maximum frequency of 700MHz (in our application 600 MHz) and the CPU is based on the C64+ core which can perform up to eight (8×8-bit) MACs per clock cycle. However, due to the high precision required in some algorithms a combination of 8×8-bit and 4×16-bit MACs have been used [8].

This processor also provides a Video Subsystem (VPSS) which is composed of a Video Front End and Video Back End. This provides a glueless interface to standard video decoder and a On Screen Display (OSD) hardware (see Fig. 5). The combination of the VPSS and the Enhanced Direct Memory Access (EDMA), simplifies considerably the data transfer from the camera to the processing buffer and from the output buffer to the display.

6. SYSTEM IMPLEMENTATION

6.1. System Overview

First, it is worth noting that the image content above the vanishing line, which does not include useful information, is not processed in order to reduce the computational complexity. The implementation of the system can be separated into three main parts as shown in Fig. 1:

- Edge detection and vanishing point voting. (Takes 16% of the total computation power)
- Multiple vanishing points detection based on the accumulator. (Takes 2% of the total computation power)
- Digital interpolation and feature extraction. (Takes 82% of the total computation power)

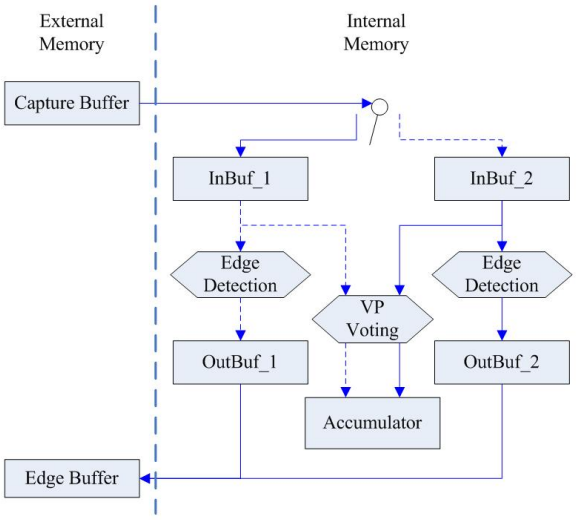


Fig. 6: System flow diagram of edge detection and vanishing point voting process.

This section describes each of these parts in details. The comparison of edge directions are not included in the implementation due to the calculation of the orientations of the gradients involves division operations, which are highly inefficient and should be avoided if possible.

The input video frame buffers are allocated by the mini-driver (the FVID module) [9] in the external heap memory. The captured frames (our test videos are in NTSC format) are then resized to 352×240 to reduce the computation. This operation is accomplished by the on-chip resizer hardware. The resizer is capable of generating output images from $1/4\times$ to $4\times$ the original image size in increments of 256/N where N is an integer between 64 and 1024 [10].

On the memory side, the 48K L1 data RAM is utilised by storing frequently used data and the internal buffers. The remaining 32K L1 data RAM and the L2 RAM are all used by cache. All the external memory used is set to be cachable.

6.2. Edge detection and vanishing point voting

While implementing this part, the most time consuming operation is the calculation of the intersections between the vanishing line and the perpendicular lines of the gradient orientations as it involves divisions. Since the dynamic range of x and y direction gradient magnitude is large and the required precision is high, a simple look-up table is not the best choice either. Instead, the 'IQmath' library [11], which is a highly optimised virtual floating point Engine, is used. This library handles the divisions by using the Newton-Raphson technique [12]. By doing so, the computational time for this part of the system is reduced by approximately 50% while keeping the same precision.

On the system level view, it is very inefficient for the CPU

to fetch data from external memory. Regardless of whether the cache is used, if the required data are not adjacent, the efficiency of the cache will be degraded. This determines the cache to constantly read and write data from and to the external memory and introduces large overhead. In this case, the use of internal data memory is crucial for the final system performance. As the L1 data RAM is limited in size, only a certain number of image lines can be transfered into the internal memory at a time. Thus, the image is divided into multiple image strips. Each time, only a few horizontal lines of image pixels are transfered into the internal memory and the processed results are then transfered back to the external memory.

The EDMA is used for the data transfer to reduce the CPU load. If the EDMA and the CPU are synchronised perfectly, the memory transfer overhead will be reduced to minimum. In our case, a double buffering mechanism is applied as shown in Fig. 6. The rectangular boxes represent the memory buffers and the hexagons represent the algorithms. The solid and dashed lines are showing different processes in time. Same line type means that these processes are in parallel. As the figure shows, once the EDMA is filling up InBuf_2, the data in InBuf_1 are being processed at the same time and vice versa. After the data in both of these buffers are processed, the edge detection results are transfered back to the external Edge Buffer for further processing. The accumulator of the vanishing points are placed in the internal memory as it will be needed for vanishing point detection.

6.3. Vanishing Point Detection

This part of implementation is relatively simple. The data in the accumulator buffer are smoothed by a 20 tap averaging filter. Following this, the vanishing points for different image sections are detected as described in Section 3. As this process is not repeated and the accumulator is stored in the internal memory, the processing time for this part is not contributing significantly overall.

6.4. Feature Extraction

The digital interpolation and feature extraction part consumes comparably much higher processing power, since interpolation, edge detection and edge compare are processed multiple times for each frame. If the memory management is not taken into account, the frame rate will drop to less than 3 fps. The image still needs to be processed strip by strip. The zooming area of the currently processed strip needs to be calculated and transfered into internal memory for further processing.

As shown in section 4 the iterative zooming process normally requires 10 steps. This indicates that in addition to the edge map of the original image, 10 different zooming areas need to be transfered into the internal memory for interpolation and edge comparison. As the maximum zooming ratio is set to be 0.9 in normal cases, the selected zooming areas have almost the same size as the original image. Therefore a large mount of data need to be transfered.

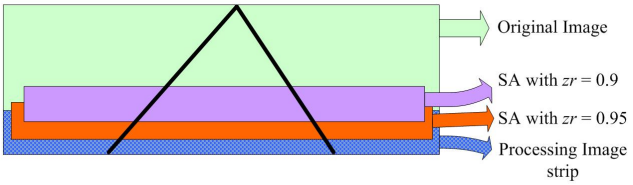


Fig. 7: Illustration of selected zooming area with different zr. The shaded block is the currently processed image strip

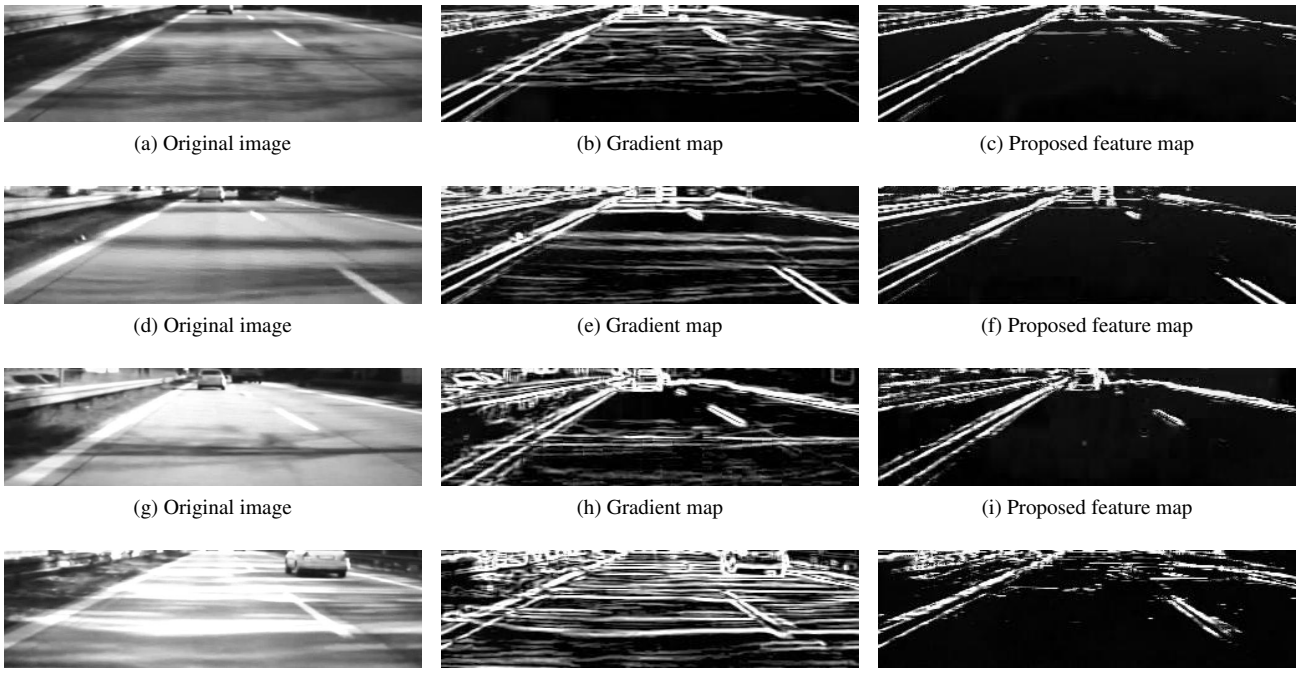
Although 10 zooming areas must be transfered, a very large portion of the image is repeated for each pair of the adjacent selected areas as shown in Fig. 7. Reusing the repeated areas will save the memory transfer significantly. The data already transfered do not need to be transfered again. An internal circular buffer is used to achieve this.

7. EXPERIMENTAL RESULTS

In this section, we show experimental results obtained using our proposed lane feature extraction algorithm. These results are summarised in Fig. 8 and they show frames captured from the output of the DSP platform. The original input images are shown in the first column. The corresponding gradient maps and the extracted feature maps are shown in the second and third column respectively. As Fig. 8 shows, all original input frames include heavy shadows and Fig. 8j contains other vehicles on the road in the near field. The gradient map is very noisy and contains a large number of unwanted features. The proposed algorithm successfully removes most of the noise and the lane features are preserved. This indicates that the fixed point implementation of the system agrees with our theoretical development. To produce these results, 10 iterations of bilinear interpolation were performed. The chosen zooming ratio zr is from 0.90 to 0.99 in increments of 0.01.

8. CONCLUSION

In this paper, a novel lane edge feature extraction algorithm based on digital interpolation is presented. The algorithm uses the global shape information of the lanes in order to extract only the lane edges in severe noise environment. A multiple vanishing points voting process is also introduced to cope with the curved lanes. The algorithm was successfully implemented and tested on the TMS320DM6437 DSP platform. The implementation results agree with the theory and were showed to be robust. The complete system is running in real-time at above 23 fps without the need of coding in linear assembly or assembly but only managing the memory efficiently.



(a) Original image (b) Gradient map (c) Proposed feature map

(d) Original image (e) Gradient map (f) Proposed feature map

(g) Original image (h) Gradient map (i) Proposed feature map

(j) Original image (k) Gradient map (l) Proposed feature map

Fig. 8: Experimental results captured from the output of the DSP platform. The first column shows the original images. The second column shows the normal gradient map of the corresponding images. The third column illustrates the feature maps obtained using the proposed feature extraction algorithm.

Further optimisation could be achieved by identifying the slow parts of the code and optimise these parts using intrinsics, linear assembly or assembly. The possibility of using the on-chip resizer to interpolate the zooming areas needs to be further investigated.

9. REFERENCES

[1] Younguk Yim and Se-Young Oh, "Three-feature based automatic lane detection algorithm (TFALDA) for autonomous driving," in *International Conference on Intelligent Transportation Systems, Proceedings*, 1999, pp. 929–932.

[2] K. Kluge and S. Lakshmanan, "A deformable-template approach to lane detection," in *Proceedings of the Intelligent Vehicles '95 Symposium*, 1995, pp. 54–59.

[3] C. Kreucher and S. Lakshmanan, "LANA: a lane extraction algorithm that uses frequency domain features," *Robotics and Automation, IEEE Transactions on*, vol. 15, pp. 343 – 350, 1995.

[4] J.M. Collado, C. Hilario, A. de la Escalera, and J.M. Armingol, "Detection and classification of road lanes with a frequency analysis," in *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, 2005, pp. 54–59.

[5] Yue Wang, Eam Khwang Teoh, and Dinggang Shen, "Lane detection and tracking using B-snake," *Image and Vision Computing*, vol. 22, pp. 269–280, 2004.

[6] Chenyang Xu and J.L. Prince, "Gradient vector flow: a new external force for snakes," in *1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Proceedings.*, 1997, pp. 66–71.

[7] Jinyou Zhang and H.-H. Nagel, "Texture-based segmentation of road images," in *Proceedings of the Intelligent Vehicles '94 Symposium*, 1994, pp. 260– 265.

[8] Naim Dahnoun, *Digital Signal Processing Implementation: using the TMS320C6000 processors*, Prentice Hall PTR, 2000.

[9] Texas Instrument, *The TMS320DM642 Video Port Mini-Driver*, 2003.

[10] Texas Instrument, *TII DM6437 VPSS Drivers: Resizer Design Specifications*.

[11] Texas Instrument, *TMS320C64x+ IQmath Library User's Guide*.

[12] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge University Press, 2nd edition, 1992.